Interactive Graphics Final Project

Introduction. I am interested in the medical application of Engineering. Therefore, I chose to follow my passion to make this project. There are about 83 millions deaf people in the world and only a few non-deaf people who know sign language. These people communicate by moving their hands and arms. A single hand is used only to express the name and surname by spelling it. All other words are associated with a gesture that is performed with the use of both hands and arms. The intent of my work is to provide an easy method of learning the sign language alphabet. For this purpose, a hand will show you, using the sign language, the name and/or surname that the user will type on the keyboard.

Three JS. For this project I used a JavaScript library called *Three*. I chose this library because, as described further in the following paragraphs, it has a function for the animation called *Tween.* Since in the previous homework I used the *setInterval()* function I preferred to use it because I found them a little bit similar to each other. Using Three.js there are 3 fundamental components:

- The *scene*
- The *camera*
- The *renderer*

The scene defines the space where everything will happen. Therefore I created it with the function *Scene()*. But in order to see what is going on a camera is needed, thus I set the camera as a child of the scene. Among the many type of cameras, I chose a perspective one (*PerspectiveCamera()*) and I set the field of view, the aspect ratio, and the near and far clipping plane. The position of the camera can change but it will always look at the position (0, 0, 0) because I fixed it using the function *lookAt()*. In this way the user can "walk around" the hand but it will never exit the view volume. In the end the renderer is needed to render the scene and all its children.

After these three fundamental components there are other two things to add:

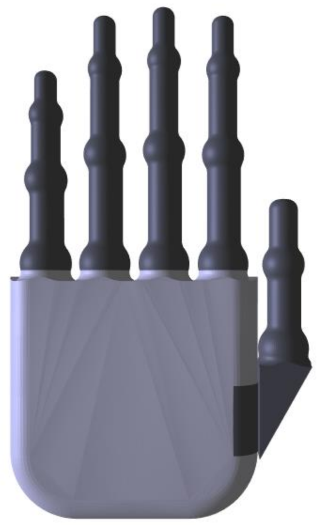
- An *object*
- The *light*

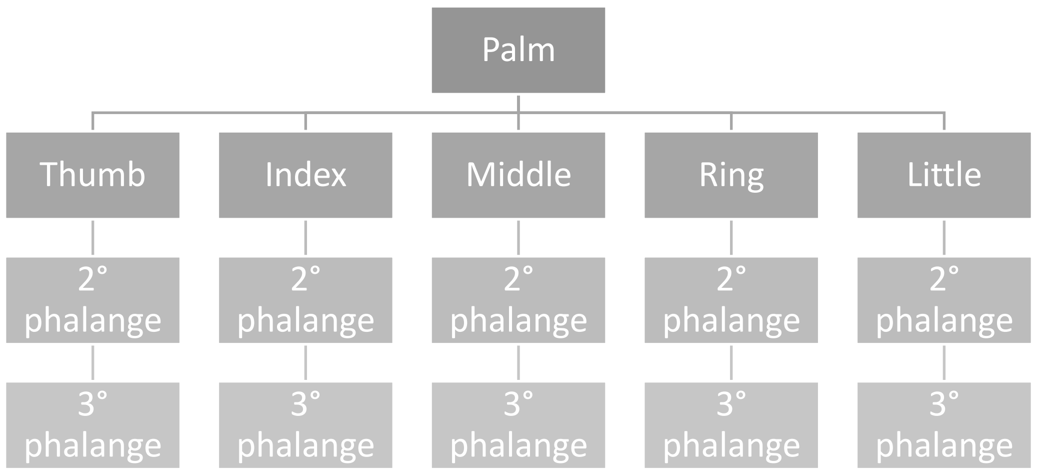
As for the camera, also these two components are added to the scene as children. I will talk about the object in the next paragraph. After adding an object, a light is needed to be able to see it; otherwise it would be like staying in a full room with the light off. I used two different lights:

- Ambient light
- Directional light

The ambient light is a global illumination of the scene. I used the function *AmbinetLight()* to create the light and to set the colour (white) and the intensity (0.5). To highlight the three-dimensionality of the object I used a directional light. This light is infinitely far away and the rays it produces are parallel to each other. With the function *DirectionalLight()* I created the light, I set its colour (light lilac) and its intensity (1.0). Then I placed the light with the function *position.set()* and I set the target's position with the function *target.position.set()*. In this way the direction of the light is calculated as pointing from the position to the target.

The model. Using *Fusion 360* I made all the components of the hand (palm and phalanges), then using *Blender* I built the hierarchical model of the hand, as showed on the graph below. To obtain a more realistic freedom of movement of the fingers I chose to reproduce the knuckles as spheres. Each phalange is attached to the previous knuckles. Thanks to this choice I was able to move the phalanges (and then the fingers) in all directions simply by rotating the spheres. To upload the model I used a Three function called *GLTFLoader()* which created a loader, than I recalled the gltf model of the hand I previously made, and in the end I loaded the model adding it to the scene. Within the loading function I also recalled the *components()* function which assigns a variable to each components of the hand. To do this I used the *getObjectByName()* function which take an object by the name that I assigned to each component when I made the model. Also inside the loading function I assigned the base dark/light grey textures to the model, as showed in the image below, but I will further discuss this topic in the next paragraph.





Colours' importance. In this project colours are not random, as showed in the background texture, there are two mains colours: blue and purple. To obtain a better understanding of the movements by the user I chose to underline, with different colours, when the hand is reproducing the name and when the surname. This difference is immediately showed on the text box where the user types the name/surname. In fact, when they type in the name box the text will appear blue, while in the surname box it will be purple. The user can see this change on the different colour of the fingers, in fact the textures will be blue or purple based on what the hand is reproducing. To upload the textures I used the function *TextureLoader().load()*, then using the function *MeshPhongMaterial()*, which uses a Phong model for calculate reflectance, I assigned these textures to the respective parts of the hand.

Palm texture

Base fingers texture

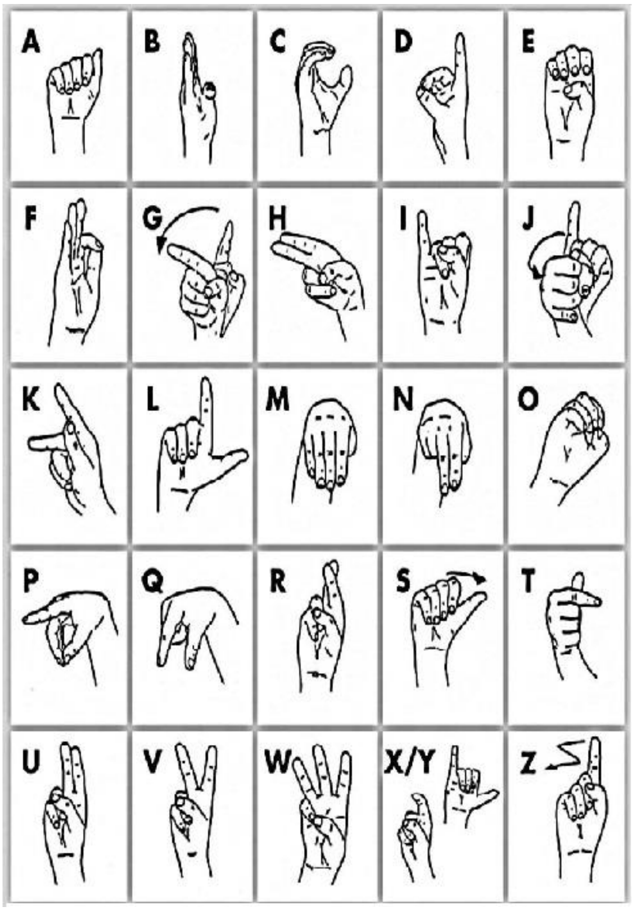
Name fingers texture

Surname fingers texture

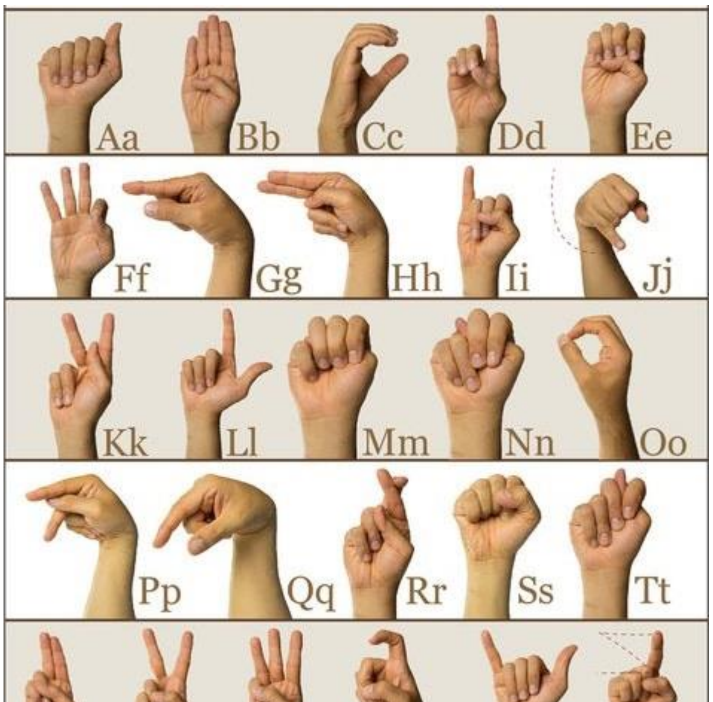
Animations. To make the animations I used *Tween.js*. Thanks to this function it is possible to change the values of the properties of an object in a smooth way. As I said before, to move the phalanges I chose to change the rotations angle of the knuckles. For each movement I set the initial rotation angle with *rotI = {}* and the respectively final rotation angle with *rotF = {}*. Then with the function *Tween(rotI).to(rotF, time)* I said that I want to go from *rotI* to *rotF* in *time* milliseconds. The tweening engine will take care of finding the intermediate values and it will do it in a linear way. To find the final rotation angles I used a slider, deleted at the end. In this way I was able to change the rotation angle and when I was satisfied with the results I read from the console the new value and I set it. Another functions that I used are *repeat(1)* and *yoyo(true)*, using together these functions I was able to "rewind" the animation so that the hand returns to the initial position. In some animation I also needed to use the *delay(times)* function so that the animation will start after *times* millisecond, and the *animation1.chain(animation2)* function which says that the *animation2* will start once *animation1* has finished. In the end to get started the animation the function *start()* is needed. All the animation are in a switch function called *animation(letter, position)*. After the user sends the name a function called *readName()* reads the first letter of the name and says to *alphabet* that it has to execute the *letter* animation which is on the position 0 of the string. When the animation is finished (*onComplete* function) the *position* is incremented by one and a function *fun(position)* is called. This function is very important because it says that if the name is still not finished *alphabet* has to execute the next letter's animation, instead the program would start to read the surname. The function *readSurname()* reads the first letter of the surname and says to *alphabet* that it has to execute the *letter* animation which is on the position 0 of the string. The *fun* function is important because without it the program will try to start all the animations together, instead of executing one animation at the time. Another important device is to set a flag to prevent the program from running animations before the previous ones are finished, otherwise the program will execute all them at the same time by superimposing them. As for spoken languages there exist a lot of sing languages. In this project I chose to use two of them:

- Italian
- American

I chose the Italian because I live in Italy ad it is my first language, and American because it is the most common.



Italian alphabet



American alphabet

Sounds. To obtain a better understanding of the movements by the user I chose to underline which letter the hand is doing by adding a sound for each letter. I used the function *AudioListener()* to create a listener and I add it to the camera and then for each sound I loaded the audio file. As for the animation, the sound needs an input to start, therefore at the beginning of each animation on the *alphabet()* function there is a *sound.play()* function. When the animation is finished I also set to stop the audio with the function *sound.stop()*. Despite this function sometimes the console finds an audio error and the animation does not work properly.

Interaction. The first interaction on this project is the choice of the language, using two buttons the user can select Italian or American. The default setting is the Italian language and when the American button is clicked a flag becomes true and the *alphabet* function will execute the animation on the American language. If then the user clicks on the Italian button the flag returns false and the *alphabet* function will execute the animation on the Italian language. The second interaction is the name and/or surname input. The user can type both or only one. It is not key sensitive, the letters like à é è ì ò ù are reading like a e i o u. If the user makes a mistake and inserts a number it will not be a problem, the *alphabet* function will jump that character and reads the next one. If the user has more than one name/surname it is not a problem, *alphabet* is able to read the space input. The third interaction is the send button which starts the animation. The fourth interaction is the possibility to change the velocity of the execution of the animations. Using a slider a parameters *t,* which multiply the *times* on the tween function, is modified and therefore the time of execution is changed. The fifth interaction is the possibility to change the volume of the audio, as for the velocity a parameter *vol* modify the *setVolum(vol)* function which set the volume. The last interaction is, as I said before, the possibility to move the camera, in this case the slider will change the value of the angle of rotation of the position of the camera, the hight does not change.

Future applications. I wish that this project will be only a start point for a more complex and sophisticated work. In the future I will amplify this work adding a bust with arms and hands and make a complete oral-sign dictionary. Maybe, after other exams, it will be interesting to change the input and use brain signal instead of keyboards input.